Building Free Browser Games from Scratch

A Practical Guide for Indie Developers

by Marvin Tang · phyfun.com

HTML5 · Cocos Creator · No-Download Games · Monetization

Free to read and share · phyfun.com

Table of Contents

Chapter 1 — Why Browser Games?

Chapter 2 — Choosing Your Tech Stack

Chapter 3 — Core Game Design Principles

Chapter 4 — Building & Shipping Your First Game

Chapter 5 — SEO & Getting Discovered

Chapter 6 — Monetization Without Ads Ruining UX

Chapter 7 — Distributing Across Platforms

Conclusion & Resources

Chapter 1

Why Browser Games?

Browser games occupy a unique space in the gaming world. They are frictionless by design — a player clicks a link and is inside your game within seconds, with no installation, no account creation, and no payment required. This zero-friction model is one of the most powerful distribution advantages any developer can have.

The case for browser games in 2024 and beyond:

- **Instant play** — no barrier between a curious visitor and your game
- **Cross-platform by default** — one codebase runs on desktop, tablet, and mobile
- **SEO-friendly** — your game lives on a URL that Google can index and rank
- **Lower development cost** — HTML5/WebGL is mature and well-documented
- **Viral potential** — sharing is just sharing a link

The Trade-offs

Browser games do have limitations. They cannot match the visual fidelity of AAA PC or console titles. Persistent storage requires either backend services or localStorage. Mobile Safari and Chrome occasionally lag behind the latest WebGL features. However, for casual and puzzle games — the most popular genres online — none of these constraints are meaningful blockers.

> ■ *Tip: The best browser games are not trying to compete with Steam titles. They win by being immediately accessible to audiences who would never open a launcher.*

Chapter 2

Choosing Your Tech Stack

For most indie developers targeting browser games in 2024, the decision comes down to three realistic options: raw HTML5/Canvas, Phaser, or Cocos Creator. Each has a distinct profile.

Option A — Raw HTML5 Canvas

Best for tiny games, prototypes, and developers who want full control. No build step, no dependency management. You write JavaScript directly against the Canvas API.

```
// Minimal game loop const canvas = document.getElementById('game'); const ctx = canvas.getContext('2d'); function loop() { ctx.clearRect(0, 0, canvas.width, canvas.height); // draw your world here requestAnimationFrame(loop); } requestAnimationFrame(loop);
```

Option B — Phaser 3

A mature, battle-tested HTML5 game framework. Excellent documentation, large community, and strong physics integration. Works well for 2D arcade and platformer games. The output is a self-contained JavaScript bundle you host anywhere.

Option C — Cocos Creator (Recommended)

Cocos Creator is the engine used to build all games on PhyFun.com. It offers a visual scene editor, a TypeScript-first workflow, and a single codebase that targets Web, iOS, Android, and WeChat Mini Games. For developers who plan to publish across multiple platforms, Cocos Creator is the most time-efficient choice.

Key advantages of Cocos Creator:

- Built-in physics (Box2D and Bullet)
- Asset pipeline with texture compression and sprite atlas packing
- One-click web build that outputs optimised HTML + JS bundles
- Free and open source — no royalties on revenue
- Strong support for WeChat and Douyin mini game export

> ■ *Tip: Start with Cocos Creator version 3.x. The 2.x branch is legacy and not recommended for new projects.*

Chapter 3

Core Game Design Principles

A technically polished game that fails on design will never retain players. Before writing a single line of code, internalise these principles.

The Five-Second Hook

A player forms their first impression within five seconds of loading your game. Your opening screen must immediately communicate what the game is and what the player should do. Avoid loading screens, tutorials, or menus that delay this moment.

Difficulty Curve, Not Difficulty Wall

The most common reason players abandon a game is not boredom — it is frustration caused by a sudden spike in difficulty. Your early levels must be completable without instruction. The player should feel clever, not confused.

A healthy difficulty progression:

Level Range	Goal	Expected Completion Rate
1 - 5	Teach core mechanic	95%+
6 - 15	Introduce variations	80%
16 - 30	Combine mechanics	60%
31+	Mastery challenges	30%

Feedback is Everything

Every player action needs an immediate, satisfying response. A button press with no visual or audio feedback feels broken. Sound effects, particle bursts, and screen shake are not decorative — they are the language your game uses to communicate with the player.

One Core Mechanic, Done Well

Resist the urge to add features. The best browser games are built around a single mechanic executed with precision. Arrow Slide Puzzle on PhyFun has one rule: slide arrows to direct a path. That simplicity is a feature, not a limitation.

Chapter 4

Building & Shipping Your First Game

Getting from idea to live URL is the most important skill an indie developer can develop. Many developers are stuck in an endless loop of building but never shipping. This chapter is a practical playbook to break that cycle.

The Two-Week Rule

For your first browser game, commit to shipping something playable within two weeks. The game does not need to be perfect. It needs to exist at a public URL. Everything else is iteration.

Project Structure (Cocos Creator)

```
assets/ scenes/ ← one scene per game state scripts/ ← TypeScript components
prefabs/ ← reusable GameObjects audio/ ← sfx and music textures/ ← sprites and
atlases build-templates/ web-mobile/ ← customise your HTML shell here
```

The Minimal Viable Game Checklist

- ✓ Core mechanic works and feels responsive
- ✓ Start screen with a clear Play button
- ✓ Win and lose states are unambiguous
- ✓ At least 10 levels or 5 minutes of content
- ✓ Mobile touch controls work correctly
- ✓ Game loads in under 3 seconds on a mobile connection
- ✓ No console errors in the browser DevTools

Hosting Options

For a personal game portal, shared hosting (such as SiteGround) with PHP and Apache/Nginx is sufficient. Your Cocos Creator web build outputs a folder of static files — upload it to a subdirectory and the game is live. For games with leaderboards or user accounts, you will need a lightweight backend.

> ■ *Tip: Use a consistent URL structure. phyfun.com/game/arrow-slide-puzzle is far better for SEO and shareability than phyfun.com/?id=1234.*

Chapter 5

SEO & Getting Discovered

A game that nobody can find is a game that nobody plays. Search engine optimisation for browser games is a genuine competitive advantage because most indie developers ignore it entirely.

Keyword Research for Game Pages

Every game page should target a specific search query. The formula is simple: [game name] + walkthrough, gameplay, or guide. Players who are stuck in a level search for exactly these terms. A game page that ranks for "[your game name] walkthrough level 1" will receive consistent organic traffic long after launch.

On-Page SEO Essentials

- **Title tag**: include the game name and a primary keyword
- **Meta description**: describe the game in 155 characters with a call to action
- **H1**: one per page, matching your title tag intent
- **Image alt text**: describe screenshots accurately
- **Internal links**: link related games to each other

Building Backlinks for a Game Site

Backlinks are the most powerful off-page SEO signal. For browser game sites, the most practical backlink sources are video platforms (YouTube, Dailymotion, Rumble, Odysee), developer community profiles (GitHub, Crunchbase, Wellfound), and Reddit posts in relevant gaming subreddits.

Every profile on a high-authority platform should include your game site URL in the website field. These are low-effort, permanent backlinks that compound over time.

Technical SEO for Game Sites

- Ensure all game pages return HTTP 200 (not soft 404)
- Submit a sitemap.xml to Google Search Console
- Use canonical tags to prevent duplicate content
- Serve your site over HTTPS
- Compress images and enable browser caching

Chapter 6

Monetization Without Ruining UX

Monetising a free browser game is a balance. Too few ads and the project is unsustainable. Too many and players leave within seconds. The goal is to generate revenue from players who enjoy the game, not at the expense of the play experience.

Display Advertising (Google AdSense / AdMob)

The most common monetisation model for browser game portals. Place one banner above or below the game canvas. Avoid interstitials on level start — they destroy session length. A single well-placed banner with high viewability will outperform three poorly placed ones.

Rewarded Video (Game Distribution Networks)

Platforms such as GameDistribution allow you to integrate rewarded video ads — the player chooses to watch a 15-30 second ad in exchange for an extra life or hint. This model is player-friendly and generates higher CPMs than display advertising.

Affiliate Links

If your site covers multiple games, affiliate links to app store listings or game merchandise are a low-friction revenue stream that does not interrupt gameplay.

What Not to Do

- Auto-playing video ads with sound
- Pop-ups that block the game canvas
- Ads that reload and flash during gameplay
- More than one interstitial per session

■ *Tip: Prioritise session length over ad impressions per session. A player who spends 20 minutes on your site and sees 3 ads is worth more than a player who sees 5 ads and leaves after 2 minutes.*

Chapter 7

Distributing Across Platforms

A browser game built with Cocos Creator can be distributed far beyond your own website. Each additional distribution channel is a new source of players and, in some cases, revenue.

Your Own Website (Primary Channel)

Your game portal is the only platform where you have full control over monetisation, branding, and SEO. It should always be your primary distribution channel.

Game Portals (CrazyGames, GameDistribution, Y8)

Submitting to established game portals provides instant access to large audiences. CrazyGames attracts over 30 million monthly players. GameDistribution acts as both a portal and a monetisation network, sharing ad revenue with developers.

Mobile App Stores

Cocos Creator exports to Android (APK) and iOS (Xcode project) from the same codebase. For markets such as Xiaomi GetApps, OPPO App Market, and Samsung Galaxy Store, you can publish without a business license as an individual developer.

WeChat & Douyin Mini Games (China Market)

If your target audience includes Chinese players, WeChat Mini Games represent a massive distribution channel. Cocos Creator has first-class support for both platforms. The process requires a Chinese business entity for WeChat, but Douyin is more accessible to individual developers.

Video Platforms as Distribution

Uploading walkthrough videos of your games to YouTube, Rumble, and Dailymotion creates a content layer that drives search traffic back to your game pages. Include your site URL in every video description.

Conclusion & Resources

Building browser games as an indie developer is one of the most rewarding creative pursuits available today. The tools are free, the distribution costs are negligible, and the audience is global. The most important thing is to ship. A live game with ten players is infinitely more valuable than a perfect game that only exists on your hard drive. Iterate in public, listen to players, and build consistently.

Useful Resources

- **PhyFun.com** — Free physics-based browser games (author's portfolio)
- **docs.cocos.com** — Official Cocos Creator documentation
- **phaser.io/tutorials** — Phaser 3 getting started guides
- **gamedistribution.com** — Game portal and SDK documentation
- **crazygames.com/developers** — Developer submission portal
- **ahrefs.com/blog** — SEO learning resources
- **r/gamedev** — Reddit community for game developers

This guide is free to share. If you found it useful, visit phyfun.com to play the games that inspired it.